IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2020.Doi Number

An Efficient Commutative Encryption and Data Hiding Scheme for HEVC Video

Bo Guan[1], Dawen Xu *[1], Member, IEEE, Qian Li[2]

[1]School of Electronics and Information Engineering, Ningbo University of Technology, Ningbo, 315211, China
[2]College of Digital Technology and Engineering, Ningbo University of Finance and Economics, Ningbo, 315175, *China*

* Corresponding author: Dawen Xu E-mail: dawenxu@126.com

ABSTRACT In this paper, an improved commutative encryption and data hiding scheme for bit rate preservation of High Efficiency Video Coding (HEVC) is proposed. To achieve commutative property, one set of syntax elements within the HEVC standard is utilized for data hiding, while another set is exploited for encryption. Specifically, the syntax elements including the sign of quantized transform coefficient (QTC), the sign of motion vector difference (MVD), and the magnitude of MVD are encrypted to perceptually distort the video. On the other hand, an improved QTC modification method is employed for data hiding to increase the embedding capacity. The proposed framework produces the format compliant bitstream and allows the hidden information to be extracted regardless of the video being in the encrypted or decrypted from. Experimental results demonstrate that the proposed scheme keeps the bit rate unchanged while increasing the embedding capacity.

INDEX TERMS Video encryption, Video data hiding, Commutative encryption and data hiding, High efficiency video coding (HEVC)

I. INTRODUCTION

With the availability of high-speed broadband access and intelligent terminal devices, video data has become the most universal and popular application carrier. People can easily access, store and transmit digital video, but the security problems are also facing serious challenges. How to ensure the content security of digital video and prevent the information contained in digital video from being eavesdropped, intercepted, destroyed or tampered by illegal users has become an important research issue in the field of cyberspace security. Video cryptography [1] and video data hiding [2, 3] have been designed to solve these problems.

Video cryptography scrambles plaintext into ciphertext using a specific key, which is then required to restore the video. The simple way to protect video content is to encrypt the entire video stream with a secure cipher, regardless of the video coding structure defined as Naive Encryption Algorithm (NEA) [1]. The NEA algorithm has disadvantages of high computational cost during the process of encryption and decryption, and will lead to format incompatibility. Therefore, as a flexible alternative to NEA, selective encryption has attracted great attention. Its main goal is to reduce the amount of data to encrypt by encrypting only some highly sensitive elements. Selective encryption should achieve a required level of security while maintaining format compatibility. In recent years, various schemes have been proposed for compressed video. According to the syntax elements to be encrypted, the works proposed in [4] and [5] encrypt transform coefficients (texture information), while that in [6] encrypts intra-prediction modes (structure information). Sallam et al. [7] proposed a RC6-based HEVC selective encryption technique that encrypts the sign bits of Non-Zero Discrete Cosine Transform (DCT) coefficients, the remaining absolute value suffixes of DCT coefficients, the sign bits of MVD and the absolute value suffixes of MVD.

Video data hiding technique aims to embed secret message into a cover media for the purpose of secure covert communication or copyright protection. In the literature, many different data hiding algorithms have been proposed and overviews of these algorithms are also provided [2, 3]. For example, some existing systems embed the information in the uncompressed domain [8, 9]. However, lossy compression is inherently contradictory with data hiding and may result in unreliable extraction of hidden information. Consequently, a great deal of research using the compressed domain has been conducted. Data embedding in compressed domain can be achieved by exploiting intra prediction modes [10], motion vectors [11], DCT coefficients [12], and entropy coding [13]. However, these methods are mainly aimed at H.264/AVC or MPEG-2 standards. Data hiding methods for the HEVC standard are still relatively scarce. Chang et al. [14] proposed an error propagation-free data hiding algorithm for HEVC intra-coded frames. Dutta and Gupta proposed the robust watermarking frameworks that

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

embed the watermark information into I frames [15] and P frames [16] of HEVC encoded video. Later, Yang and Li [17] proposed an efficient information hiding method for HEVC by introducing motion vector space.

Although many methods of video encryption and data hiding have been proposed, there are few reports of commutative encryption and data hiding. In [18], a scheme to realize commutative encryption and watermarking is designed. In this scheme, IPM, MVD and DCT coefficients' signs are encrypted, while DCT coefficients' amplitudes are adaptively watermarked. However, the independence of encryption and embedding may lead to some security flaws. To overcome the security problem, Lian [19] proposed a quasi-commutative watermarking and encryption scheme for MPEG-2 video. In addition, some schemes of irreversible data hiding [20-23] and reversible data hiding [24-26] in partially encrypted H.264/AVC or HEVC videos are investigated.

In [27], an efficient scheme of commutative encryption and data hiding based on HEVC codec is provided. Quantized transform coefficients' (*QTCs*) signs, *MVDs*' signs and *IPMs* are encrypted, while Non-Zero *QTCs*' level values are utilized for data embedding. However, due to the encryption of *IPM*, the bit rate will be actually increased. In addition, data embedding is carried out by single *bin-string* substitution. That is to say, a qualified bin-string can only be used to hide one bit of information, which means that the substitution does not make full use of the embedding space. Aiming at these challenges, in this paper, an improved version of commutative encryption and data hiding method in HEVC compression is designed. The improvement focuses on providing more efficient selective encryption and data embedding mechanism. In order to prevent encryption and hiding from interfering with each other, the elements in the HEVC coding structure are divided into two groups. The first set of elements can lead to significant perceptual distortion even when they are slightly modified, so they are suitable for encryption. The second group is suitable for data hiding because their modifications will have little effect on visual quality. Specifically, such parameters as the sign bits of *QTC*s, the sign bits of *MVD*s, and the absolute value suffixes of *MVDs* are encrypted, while the Non-Zero *QTCs*' level is utilized for data embedding. The advantage is that it can maintain constant bit rate and format compatibility by encrypting sensitive video elements. In addition, specific coefficient modification technique is employed for data hiding, which provides higher embedding payload.

The remainder of the paper is organized as follows. In Section II, the proposed scheme is introduced. Experimental results and analysis are presented in Section III. Finally in Section IV, conclusion and future work are drawn.

II. PROPOSED SCHEME

The proposed scheme consists of two parts, i.e., video encryption and video data hiding. If encryption and data hiding operations are commutative, the following equation will be established [27].

$$\boldsymbol{S(E(O,K_e),K_s) = E(S(O,K_s),K_e)} \quad (1)$$

where $\boldsymbol{O}$ represents the original video, $\boldsymbol{E}()$ represents the encryption algorithm, $\boldsymbol{S}()$ represents the data hiding algorithm, $\boldsymbol{K_e}$ represents the encryption key, and $\boldsymbol{K_s}$ represents the data hiding key.

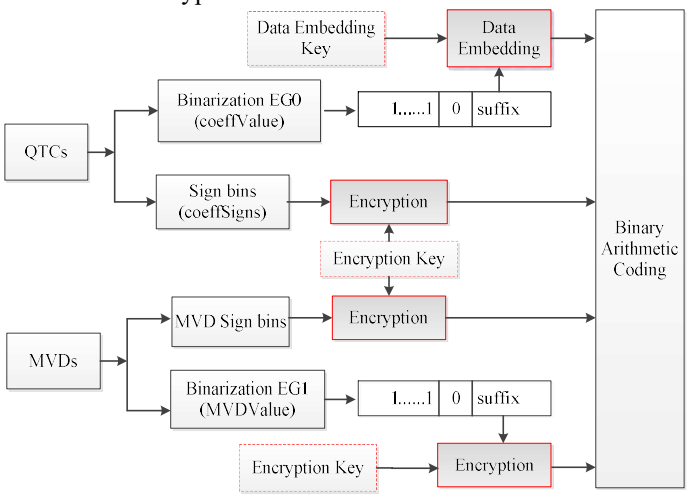


Fig.1 The Block Diagram of HEVC Selective Encryption and Data Embedding

A. Selective Encryption of HEVC Video

Context Adaptive Binary Arithmetic Coding (CABAC) is a form of entropy coding first introduced in H.264/AVC and now used in HEVC. The core algorithm of CABAC involves binarization, context modeling, and arithmetic coding [29, 30]. In the binarization step, non-binary syntax elements are converted to binary symbols (*bins*), also known as "*bin strings*". Context modeling is to estimate the probability of the various bins. Arithmetic coding refers to

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

compressing the bins to bits based on the estimated probability. Arithmetic coding can be carried out in either *regular* mode or *bypass* mode. For *regular* mode, the probability mode of the bin is selected by the context, which refers to the previously encoded syntax elements. In *bypass* mode, a fixed probability model is used to perform arithmetic coding.

Nowadays, *CABAC*-based encryption is still a challenging research topic. The reason is that the bits in CABAC compressed bitstream are very interdependent. The proposed HEVC selective encryption uses the stream cipher to encrypt a set of the bins that use the bypass-*BAC* mode in the entropy process, so as to ensure format compatibility and maintain the same bit rate. From the perspective of binarization operation, the truncated rice code and *k*-th order Exp-Golomb (*EGk*) code can meet these constraints of HEVC selective encryption. In this work, the encryption space is constituted of the sign bits of *QTC*s, the sign bits of *MVD*s, and *MVD* absolute value. The main consideration is that any modification of the MVD suffix and the sign bins will not change the format compatibility and maintain the same bit rate. Fig.1 shows the block diagram of the proposed HEVC selective encryption technique.

QTC's Sign Encryption: For each non-zero residual macroblock, a stream cipher (e.g., Rabbit or RC4) is used to encrypt the signs bits of *QTCs* in both I-frames and P-frames. Typically, a XOR operation is performed between the bits of plaintext and the secret bits generated from a pseudorandom number generator (PRNG). Consequently, the bits of cipher text can be obtained as

$$C_i = X_i \oplus K_i \tag{2}$$

where X_i represents the sign bit of the *i*-th non-zero *QTC* to be encrypted. If the *QTC* is positive, X_i is equal to 0. Otherwise, X_i is equal to 1. K_i denotes the secret bit generated by the stream cipher, and C_i is the corresponding ciphertext bit. The main advantages of this algorithm are simplicity and high speed.

More importantly, the sign bit of non-zero *QTC* is encoded in *bypass* mode. In *bypass* mode, each bin is directly encoded via the *bypass* coding engine without updating probability model, so its encryption will not affect compression performance.

MVD's Sign Encryption: The motion vectors (*MVs*) play an important role for inter-prediction in HEVC video coding, and it is very important for video reconstruction. In HEVC, advanced motion vector prediction is adopted to predict current motion vector. That is, not *MVs* but *MVDs* are actually coded in the bitstream. Encrypting *MVDs* is critical for protecting contours and motion information. Since the signs and magnitudes of *MVDs* are encoded separately in HEVC, they are encrypted separately in the proposed scheme. The signs of *MVDs* are coded in *bypass* mode which means changing zero to one or vice versa does not affect the compression ratio. In order to keep the length of the coded stream unchanged, the signs of *MVDs* are encrypted by applying the bitwise XOR operation with a stream cipher.

MVD's Magnitude Encryption: In H.264/AVC, the first -nine bins of *MVD* are regular coded truncated unary bins, followed by 3rd order Exp-Golomb bins that are coded via *bypass* mode. In HEVC, the number of regular coded bins for *MVD* is greatly reduced [31]. Only the first two bins are regular coded (abs_mvd_greater0_flag, abs_mvd_greater1_flag), followed by first-order Exp-Golomb (EG1) bins that are coded via *bypass* mode (abs_mvd_minus2). TABLE.1 shows the binarization of *MVD*'s magnitude. The *MVD*'s magnitude encryption is performed by XORing the suffix with pseudorandom bits, which is determined by an encryption key as shown in Fig. 1. In order to achieve format compatibility, the encryption should keep the length of the bin string unchanged.

The proposed encryption technology is to encrypt HEVC video content while maintaining format compatibility and the same bit rate. Format compatibility is achieved by encrypting bins in bypass mode, and the structure of HEVC video bitstream is not modified. By encrypting *QTCs* sign bits, *MVDs* sign bits, and M*VD* absolute value suffixes, it is ensured that the encrypted bin has the same length as the original bin, thereby achieving the same bit rate.

TABLE.1 BINARIZATION OF MVD'S MAGNITUDE

MVD's Magnitude	abs_mvd_minus2	Bin String
0		abs_mvd_greater0_flag
1		abs_mvd_greater1_flag
2	0	0 0
3	1	0 1
4	2	10 00
5	3	10 01
6	4	10 10
7	5	10 11
8	6	110 000
9	7	110 001
10	8	110 010
11	9	110 011
12	10	110 100
13	11	110 101
14	12	110 110
15	13	110 111
16	14	1110 0000
17	15	1110 0001
18	16	1110 0010
19	17	1110 0011
20	18	1110 0100
21	19	1110 0101
22	20	1110 0110
23	21	1110 0111
…	…	…

In summary, the proposed HEVC selective encryption scheme is to encrypt the sensitive bin strings, which has no impact on the compression ratio and can achieve effective perceptual scrambling performance. This will be confirmed by later experimental observations. In addition, only XOR operations are performed with the features of low complexity

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

overhead and fast encoding time.

B. Decryption of HEVC Video

The decryption process can be performed by an authorized user with the encryption key. According to the encryption process, we will decrypt the sign of *QTC*s, the sign of *MVD*s, and the magnitude of *MVDs* respectively. The decryption operation is relatively simple. Specifically, the encrypted bin-strings are decrypted using the same cipher stream as the encryption process. Because the XOR operation is symmetrical, the decryption operation will be the same as the encryption operation. Therefore，the detailed decryption process is omitted.

C. Data Embedding Process

To realize commutative encryption and data hiding, the syntax elements utilized for data embedding should be distinguished from the syntax elements selected for encryption. In this work, the coefficient level is considered for data embedding. In H.264/AVC, the coefficient level is coded as a truncated unary code in regular mode for bins 1 to 14. If the level is larger than 14, then a suffix is appended to the truncated unary code. The suffix is binarized with a 0th-order Exp-Golomb code (EG0) and coded in bypass mode. However, in HEVC, only the first two bins of (i.e., *coeff_abs_level_greater1_flag* and *coeff_abs_level_greater2_flag*) are regular coded [31]. The remaining level (*coeff_abs_level_remaining*) is coded in *bypass* mode in order to increase throughput. HEVC employs Golomb-Rice codes for small values and Exp-Golomb code for larger values. The Rice parameter m is set to 0 at the beginning of each coefficient group and it is conditionally updated depending on the previous value of the parameter and the current absolute level as follows [32].

If $absCoeffLevel > 3 \times 2^m$, $m = min(4, m+1)$ (3)

where the absolute value of the coefficient is calculated as

$$absCoeffLevel = baseLevel + coeff\_abs\_level\_remaining \quad (4)$$

The *baseLevel* of a coefficient is defined as

$$baseLevel = significant\_coeff\_flag + coeff\_abs\_level\_greater1\_flag + coeff\_abs\_level\_greater2\_flag \quad (5)$$

where a flag has a value of 0 or 1 and is inferred to be 0 if not present.

Generally, for the binarization of the syntax element *coeff_abs_level_remaining*, the prefix is coded using a truncated unary code, and the suffix is coded using a fixed length code. In order to improve coding efficiency, the number of fixed length bins also depends on Rice parameter m. When m ranges from 0 to 4, the binarization of the remaining level are shown in TABLE. 2 to 6, respectively [27].

TABLE.2 BINARIZATION OF *THE REMAINING LEVEL* (m=0)

coeff_abs_level_remaining	*prefix*	*suffix*	Suffix Range
0	0		
1	10		
2	110		
3	1110		
4~5	11110	x	0 to 1
6~9	111110	xx	0 to 3
10~17	1111110	xxx	0 to 7
18~33	11111110	xxxx	0 to 15
…	…	…	…

TABLE.3 BINARIZATION OF *THE REMAINING LEVEL* (m=1)

coeff_abs_level_remaining	*prefix*	*suffix*	Suffix Range
0~1	0	x	0 to 1
2~3	10	x	0 to 1
4~5	110	x	0 to 1
6~7	1110	x	0 to 1
8~11	11110	xx	0 to 3
12~19	111110	xxx	0 to 7
20~35	1111110	xxxx	0 to 15
36~67	11111110	xxxxx	0 to 31
…	…	…	…

TABLE.4 BINARIZATION OF *THE REMAINING LEVEL* (m=2)

coeff_abs_level_remaining	*prefix*	*suffix*	Suffix Range
0~3	0	xx	0 to 3
4~7	10	xx	0 to 3
8~11	110	xx	0 to 3
12~15	1110	xx	0 to 3
16~23	11110	xxx	0 to 7
24~39	111110	xxxx	0 to 15
40~71	1111110	xxxxx	0 to 31
…	…	…	…

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

TABLE.5 BINARIZATION OF *THE REMAINING LEVEL* (*m*=3)

coeff_abs_level_remaining	*prefix*	*suffix*	Suffix Range
0~7	0	xxx	0 to 7
8~15	10	xxx	0 to 7
16~23	110	xxx	0 to 7
24~31	1110	xxx	0 to 7
32~47	11110	xxxx	0 to 15
48~79	111110	xxxxx	0 to 31
…	…	…	…

TABLE.6 BINARIZATION OF *THE REMAINING LEVEL* (*m*=4)

coeff_abs_level_remaining	*prefix*	*suffix*	Suffix Range
0~15	0	xxxx	0 to 15
16~31	10	xxxx	0 to 15
32~47	110	xxxx	0 to 15
48~63	1110	xxxx	0 to 15
64~95	11110	xxxxx	0 to 31
…	…	…	…

Data embedding can be accomplished by modifying eligible *bin-strings* of *coeff_abs_level_remaining* in TABLE.2~6. Generally, in order to improve the security, the additional data is first encrypted using a data-hiding key. The encrypted version can be denoted as $W = \{w(i)|i = 1,2,\dots,K, w(i) \in \{0,1\}\}$. According to the Rice parameter value, the data embedding operation is carried out in the following five cases.

$$\overline{EG0\_SUFFIX} = \begin{cases} '0' & if\ w(i) = 0\ \&\ EG0\_SUFFIX \in C1 \\ '1' & if\ w(i) = 1\ \&\ EG0\_SUFFIX \in C1 \end{cases} \tag{6}$$

$$\overline{EG0\_SUFFIX} = \begin{cases} '00' & if\ w(i,i+1) = 00\ \&\ EG0\_SUFFIX \in C2 \\ '01' & if\ w(i,i+1) = 01\ \&\ EG0\_SUFFIX \in C2 \\ '10' & if\ w(i,i+1) = 10\ \&\ EG0\_SUFFIX \in C2 \\ '11' & if\ w(i,i+1) = 11\ \&\ EG0\_SUFFIX \in C2 \end{cases} \tag{7}$$

$$\overline{EG0\_SUFFIX} = \begin{cases} '000' & if\ w(i,i+1) = 00\ \&\ EG0\_SUFFIX \in C3 \\ '001' & if\ w(i,i+1) = 01\ \&\ EG0\_SUFFIX \in \{C3, C4\} \\ '010' & if\ w(i,i+1) = 10\ \&\ EG0\_SUFFIX \in \{C3, C4\} \\ '011' & if\ w(i,i+1) = 11\ \&\ EG0\_SUFFIX \in \{C3, C4, C5\} \\ '100' & if\ w(i,i+1) = 00\ \&\ EG0\_SUFFIX \in \{C4, C5, C6\} \\ '101' & if\ w(i,i+1) = 01\ \&\ EG0\_SUFFIX \in \{C5, C6\} \\ '110' & if\ w(i,i+1) = 10\ \&\ EG0\_SUFFIX \in \{C5, C6\} \\ '111' & if\ w(i,i+1) = 11\ \&\ EG0\_SUFFIX \in C6 \end{cases} \tag{8}$$

where $\overline{EG0\_SUFFIX}$ denotes the *suffix* of *bin-string* after data embedding. To simplify, only EG0_suffix is listed. In addition, $C1 = \{'0', '1'\}$, $C2 = \{'00', '01', '10', '11'\}$, $C3 = \{'000', '001', '010'\}$, $C4 = \{'011'\}$, $C5 = \{'100'\}$, and $C6 = \{'101', '110', '111'\}$. For example, when *coeff_abs_level_remaining* is equal to 13, the *EG0_SUFFIX* of its bin-string is "011" which belongs to *C4* . If the to-be-embedded bits are "01", the *EG0_SUFFIX* "011" should be substituted with "001" which is the corresponding replaceable bin-string in *C3*. After the substitution operation, the *coeff_abs_level_remaining* will be changed to 11.

(2) Case 2: Rice parameter *m* =1

According to Eq. (3), if the value of *absCoeffLevel* changes from 6 to 7 due to data hiding, the rice parameter *m* may fall into another region. As a result, the wrong Rice parameter will be used to decode the *absCoeffLevel.* To ensure format compatibility, data embedding will not be performed when the current *absCoeffLevel* is equal to 6 or 7. Otherwise, if the current *coeff_abs_level_remaining* value is less than or equal to 7, the data embedding procedure is presented in Fig.2. In addition, if it is in the interval [8, 11], data embedding can be performed as Eq. (7). If it is in the interval [12, 19], data embedding can be performed as Eq. (8).

(3) Case 3: Rice parameter *m* = 2

In this case, for format compatibility, data embedding is also not performed when the current *absCoeffLevel* is equal to 12 or 13. In addition, if the current *coeff_abs_level_remaining* value is in the interval [0, 3] or [4, 7], data embedding can be performed as Eq. (7). Otherwise, the data embedding will be performed according to the

(1) Case 1: Rice parameter *m* =0

As can be seen from TABLE.2, when *coeff_abs_level_remaining* is less than or equal to 3, there is no suffix that satisfies the equal-length substitution condition. Therefore, if the current value is less than or equal to 3, data embedding will be skipped. Otherwise, the secret bits can be embedded by *bin-string* substitution as follows.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

procedure in Fig.2. The only difference is to replace 8 and 5 in Fig.2 with 14 and 11, respectively.

(4) Case 4: Rice parameter m = 3

Similarly, it can be inferred from Eq.(3) that when the current *absCoeffLevel* is equal to 24 or 25, data embedding cannot be performed. In addition, if the current *coeff_abs_level_remaining* value is in the interval [0, 7] or [8, 15], data embedding can be performed as Eq. (8). Otherwise, the data embedding is performed according to the procedure in Fig.2. It should be noted that 8 should be replaced with 26 and 5 with 23.

(5) Case 5: Rice parameter m= 4

Since the coefficients satisfying this situation are scarce, data embedding is not considered.

The advantage of the algorithm is that data embedding can be carried out through simple *QTC* modification. Furthermore, the *bin-string* of the marked coefficient has the same length as the original *bin-string* and thus the length of the Network Abstraction Layer (NAL) units is preserved as well. Since I-frames are critical to the video sequence, distortion occurring in I-frames due to data hiding will propagate to subsequent P-frames. Therefore, the *QTC* modification in this paper is only implemented in P-frames, while all bin-strings of *absCoeffLevel* in I-frames remain unchanged. In general, P-frames have smaller embedding capacity because they are highly compressed by motion compensation and entropy coding.

Procedure
if (w_i = =0 and *coeff_abs_level_remianing* %2= =0)
{
The current *absCoeffLevel* is unmodified. And let *i*=*i*+1.
}
else if (w_i = =0 and *coeff_abs_level_remianing* %2= =1)
{
if (*absCoeffLevel* = =8) *absCoeffLevel*=*absCoeffLeve*l -1.
else
{ *absCoeffLevel*=*absCoeffLevel -1.* And let *i*=*i*+1.}
}
else if (w_i = =1 and *coeff_abs_level_remianing* %2= =1)
{
The current *absCoeffLevel* is unmodified. And let *i*=*i*+1.
}
else if (w_i = =1 and *coeff_abs_level_remianing* %2= =0)
{
if (*absCoeffLevel* = =5) *absCoeffLevel*=*absCoeffLeve*l +1.
else
{ *absCoeffLevel*=*absCoeffLeve*l+1. And let *i*=*i*+1.}
}

Fig.2 The procedure of data embedding

D. Data Extraction Process

As with the embedding process, data extraction can be performed in the following five categories according to the Rice parameter.

(1) Case 1: Rice parameter m = 0

Obviously, from the embedding process, if the current *coeff_abs_level_remaining* value is less than or equal to 3, data extraction will be skipped. Otherwise, data extraction is performed as follows.

$$\overline{w(i)} = \begin{cases} 0 & if \ \overline{EG0\_SUFFIX} = '0' \\ 1 & if \ \overline{EG0\_SUFFIX} = '1' \end{cases} \quad (9)$$

$$\overline{w(i,i+1)} = \begin{cases} 00 & if \overline{EG0\_SUFFIX} \in \{'00','000','100'\} \\ 01 & if \overline{EG0\_SUFFIX} \in \{'01','001','101'\} \\ 10 & if \overline{EG0\_SUFFIX} \in \{'10','010','110'\} \\ 11 & if \overline{EG0\_SUFFIX} \in \{'11','011','111'\} \end{cases} \quad (10)$$

where $\overline{w(i)}$ denotes the extracted data, $\overline{EG0\_SUFFIX}$ is the *suffix* of currently processing *bin-string* in encrypted domain.

(2) Case 2: Rice parameter m = 1

According to the previous embedding process, if the current *absCoeffLevel* value is equal to 6 or 7, data extraction will not be performed. If the current *coeff_abs_level_remaining* value is less than or equal to 7, data extraction can be performed as Eq. (9). Otherwise, if it is in the interval [8, 11] or [12, 19], the extraction of the data bit can be performed as Eq. (10).

(3) Case 3: Rice parameter m = 2

Similarly, if the current *absCoeffLevel* value is equal to 12 or 13, data extraction will not be performed. If the current *coeff_abs_level_remaining* value is in the interval [0, 3] or [4, 7], data extraction can be carried out as Eq. (10). Otherwise, data extraction is performed as Eq.(9).

(4) Case 4: Rice parameter m = 3

As before, if the current *absCoeffLevel* value is equal to 24 or 25, data extraction will not be performed. If the current *coeff_abs_level_remaining* value is in the interval [0, 7] or [8, 15], data extraction can be carried out as Eq. (10). Otherwise, data extraction is performed as Eq. (9).

(5) Case 5: Rice parameter m=4

In this case, data extraction will not be performed.

With the data-hiding key, the extracted hidden bits can be further decrypted to obtain the original message. It can be noted that in the proposed scheme, those syntax elements for encryption are not used for data embedding, and vice versa. Therefore, the order of embedding or encryption is irrelevant. Consequently, the proposed scheme conforms to the commutative property of Eq. (1).

III. EXPERIMENTAL RESULTS AND DISCUSSIONS

The proposed scheme is implemented by applying the encryption and data embedding on the HEVC reference software version HM-12.0 [33]. TABLE.7 defines the set of benchmark video sequences that are used in the performance study. These video sequences have different characteristics, such as high or low texture, high or low motion. The proposed encryption and data hiding scheme is applied to all the benchmark video sequences for *low delay* mode. The frame number is 100, and intra period is set to 4.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

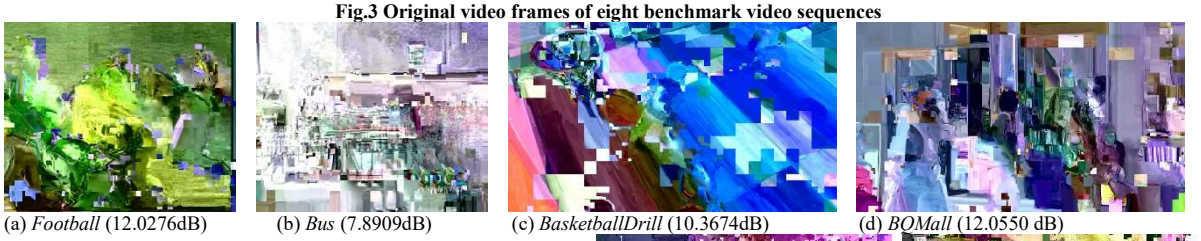
TABLE.7 THE SET OF BENCHMARK VIDEO SEQUENCES USED TO EVALUATE THE PERFORMANCE

Class	Resolution	Frame Rate	Videos
A	352×288	30	*Football*, *Bus*
B	832×480	50-60	*BasketballDrill*, *BQMall*
C	1920×1080	24	*Kimono*, *Tennis*
D	2560×1600	30	*PeopleOnStreet*, *Traffic*

(a) *Football*: the 10th frame (b) *Bus*: the 10th frame (c) *BasketballDrill*: the 10th frame (d) *BQMall*: the 10th frame

(e) *Kimono*: the 10th frame (f) *Tennis:* the 10th frame (g) *PeopleOnStreet*: the 10th frame (h) *Traffic*: the 10th frame

Fig.3 Original video frames of eight benchmark video sequences



(a) *Football* (12.0276dB) (b) *Bus* (7.8909dB) (c) *BasketballDrill* (10.3674dB) (d) *BQMall* (12.0550 dB)

(e) *Kimono* (8.4353 dB) (f) *Tennis* (7.3065dB) (g) *PeopleOnStreet* (11.0796 dB) (h) *Traffic* (10.2312 dB)

Fig.4 The corresponding encrypted video frames

A. Scrambling effect and security analysis

To quantify the visual degradation, a numerical quality analysis is provided in this section. The Peak Signal to Noise Ratio (*PSNR*), the Structural Similarity Index (*SSIM*), and the Video Quality Measurement (*VQM*) metrics are the common metrics used to evaluate video encryption [20]. The proposed scheme encrypts the signs of *QTCs*, the signs of *MVDs*, and the Magnitudes of *MVDs*, which seriously distorts video content and ensures the perception security. TABLE.8 illustrates the results of eight video sequence without encryption and with selective encryption for *QP* value 28 and 32. Average *PSNR* value for all sequences is 10.87dB for *QP* value 28, and is 10.74dB for *QP* value 32 for selective encryption. It can be noticed that the objective quality of the video is decreased to the low quality level, which confirms that the proposed algorithm can provide effective visual security.

The visualization of the encryption effect is also shown in Fig.3 and Fig.4. Fig.3 shows an original frame of each video, and Fig.4 shows the corresponding encrypted results. Due to space limitations, we do not list the visual results of all frames. It can be clearly seen that encrypted frames cannot be recognized by human vision. In Fig.5, each frame analysis is supplemented by a comparison between the original video and the corresponding encrypted video. In summary, the areas containing many details and textures will have a lot of non-zero coefficients, so it will be highly encrypted. On the other hand, uniform regions in each video frame, which contain a series of identical pixels, are less encrypted. Although the *PSNR* value of each frame has a small fluctuation, the scrambling effect can be guaranteed.

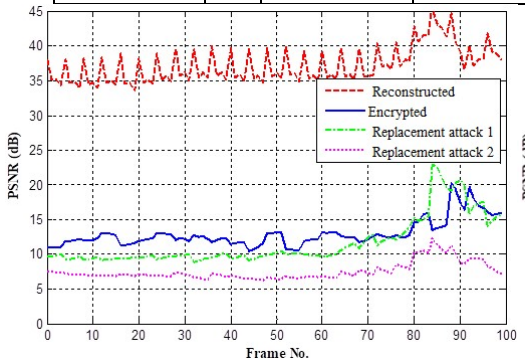
Cryptographic security of the proposed video encryption scheme depends on the adopted stream cipher. The traditional cipher is adopted in our scheme, and its security has been authenticated. One of the common attacks in video encryption is the replacement attack. In the replacement attack, the encrypted bits are usually replaced with constant bits to improve video quality and make it watchable. In our experiment, the replacement attack is performed by setting all the cipherable bits to "1" (marked as "*Replacement attack* 1") or "0" (marked as "*Replacement attack* 2") [23]. After the replacement attack, *PSNR* values of the corresponding frames

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

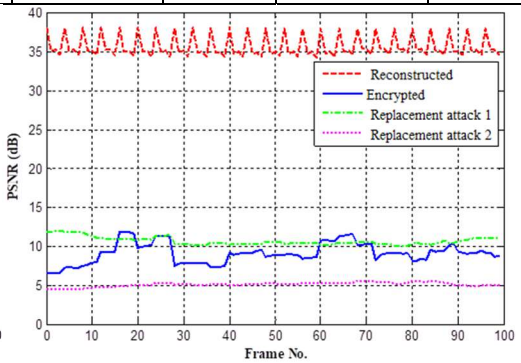
are also shown in Fig.5. It can be seen that the quality improvement is still very limited, which means that none of the sequences can provide perceptible reconstruction. This confirms that the proposed encryption scheme is robust to the replacement attack. Moreover, histogram analysis is also carried out. Ideally, the histogram of cryptographic video frame should be different from that of plain video frame [27]. This can be guaranteed in our algorithm. Taking Kimono and Tennis video as examples, the histograms of the 10th frame before and after encryption are shown in Fig.6 ~ 9.

TABLE.8 PERCEPTUAL QUALITY OF THE ENCRYPTED VIDEOS

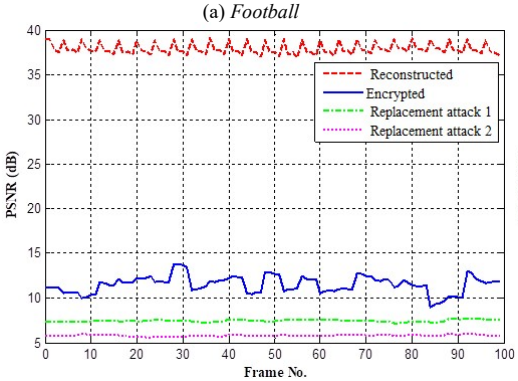
Sequence	QP	PSNR (dB)		SSIM		VQM	
		Reconstructed	Encrypted	Reconstructed	Encrypted	Reconstructed	Encrypted
Football	32	34.2510	12.8877	0.9004	0.3288	1.1290	9.9598
	28	37.1813	12.9634	0.9434	0.3228	0.8573	9.9069
Bus	32	32.8120	9.3961	0.9328	0.2183	1.2732	10.2807
	28	35.6082	9.1046	0.9606	0.2095	0.9743	10.4647
BasketballDrill	32	35.6245	11.8614	0.9013	0.3757	1.0704	12.0096
	28	37.8842	11.5383	0.9342	0.3487	0.8449	12.8574
BQMall	32	35.2199	12.1766	0.9272	0.3511	1.0752	10.1694
	28	37.5068	11.6471	0.9498	0.3413	0.8566	9.9966
Kimono	32	38.6690	10.5612	0.9337	0.4545	0.9270	8.9011
	28	40.5499	10.3708	0.9527	0.4428	0.7684	9.1386
Tennis	32	37.9429	8.5510	0.9189	0.3746	0.9440	9.4087
	28	39.7046	9.9851	0.9436	0.3912	0.7844	9.0739
PeopleOnStreet	32	35.9914	10.2785	0.9261	0.3622	0.9739	11.1340
	28	38.3239	10.7466	0.9498	0.3595	0.7723	10.6729
Traffic	32	37.1660	10.2398	0.9399	0.3667	1.0257	12.2554
	28	39.0604	10.6305	0.9581	0.3600	0.8420	12.3852



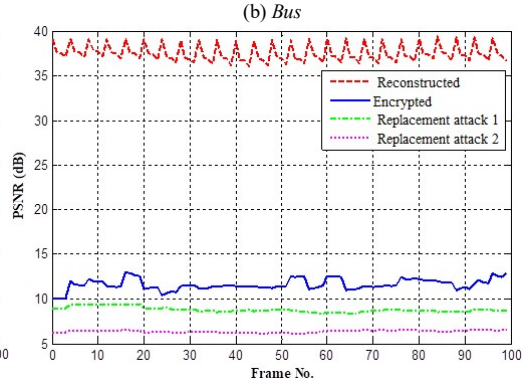
(a) *Football*



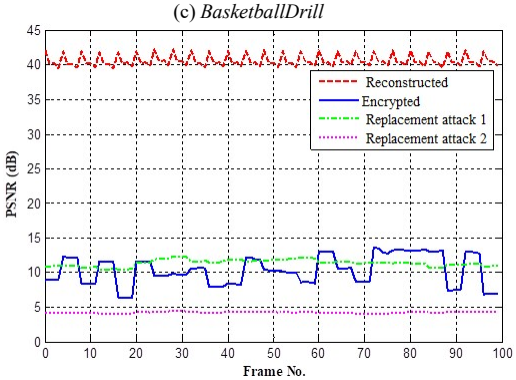
(b) *Bus*

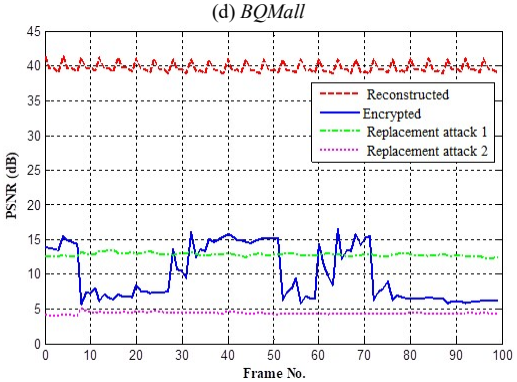


(c) *BasketballDrill*

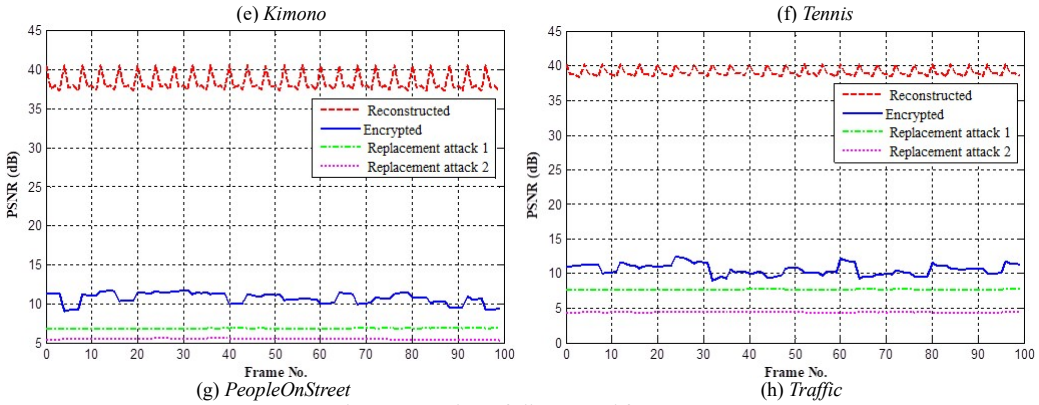


(d) *BQMall*





This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access



(g) *PeopleOnStreet* (h) *Traffic*

Fig.5 PSNR values of all encrypted frames

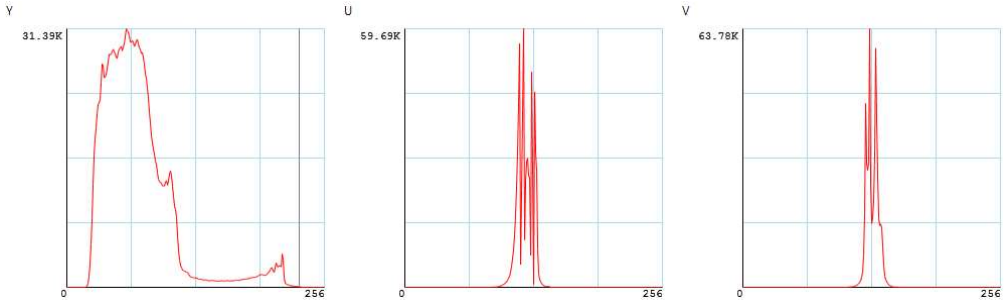


Fig.6 Histogram of *Kimono* plain video Frame #10

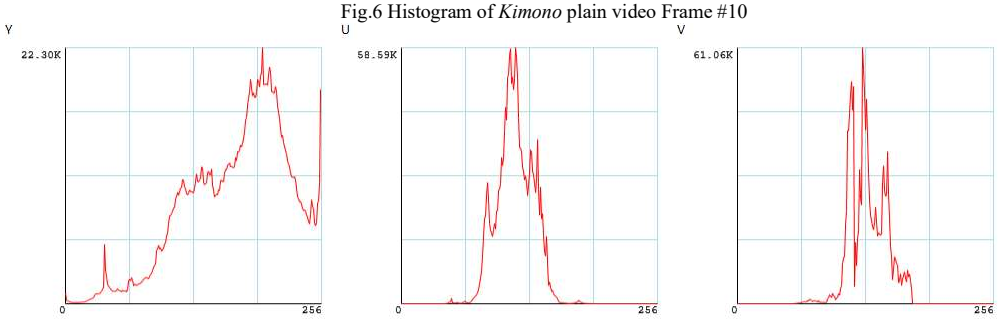


Fig.7 Histogram of *Kimono* cipher video Frame #10

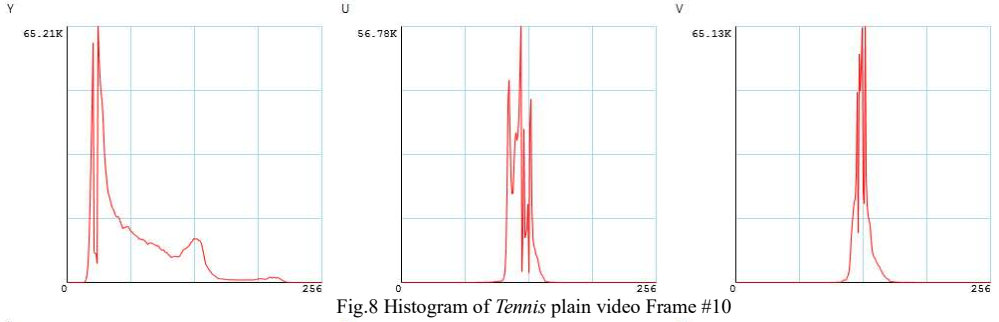


Fig.8 Histogram of *Tennis* plain video Frame #10

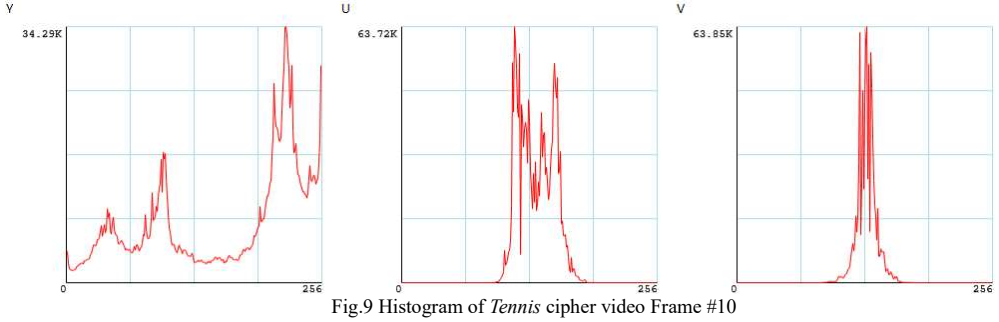


Fig.9 Histogram of *Tennis* cipher video Frame #10

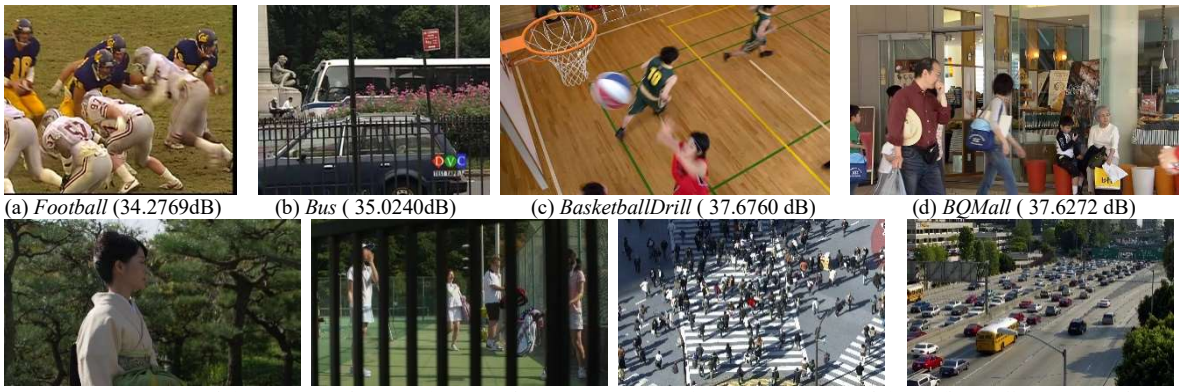
B. Visual Quality of Decrypted Video

In some cases, encrypted video containing hidden data

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

provided by the server needs to be decrypted by an authorized user. To evaluate visual quality, the corresponding decrypted versions containing hidden data are shown in Figures 2 and 10, respectively. It can be seen that the perceptual quality of marked video is almost the same as that of original video.

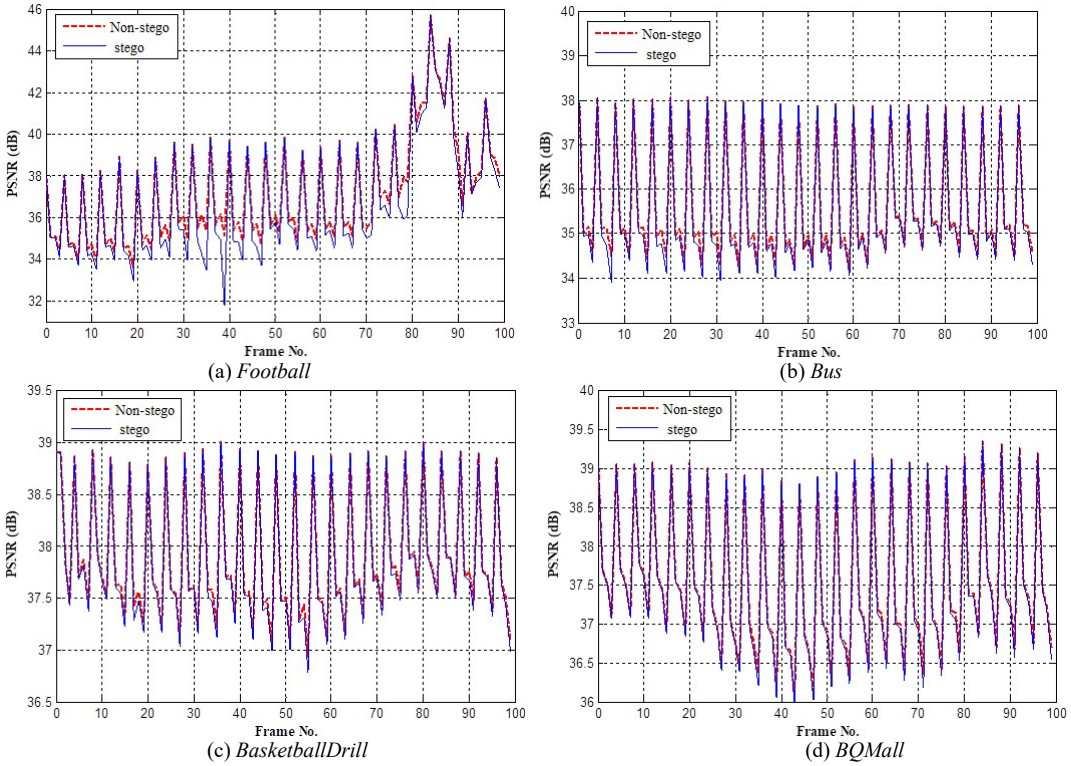
Since HEVC is lossy compression, video perception quality is degraded even without data hiding and encryption. Generally, the larger the *QP* value, the greater the distortion. In order to better demonstrate the impact of data embedding on video quality, the visual quality of un-marked video stream should also be tested. Specifically, the video sequence obtained by decompressing the un-marked video stream (i.e., reconstructed video) is used as a target sequence, and the original uncompressed video sequence is used as a reference video sequence. Similarly, in order to test the visual quality of marked video stream, the video sequence obtained through encryption, data hiding, decryption and decompression processing is used as a target sequence. It can be seen that in this case, the target video contains hidden data. The comparison results are listed in TABLE.9. The visual quality degradation caused by data hiding is very low, that is, it is usually difficult to detect the degradation in video quality. Furthermore, the *PSNR* values of all marked frames are shown in Fig.11.



(a) *Football* (34.2769dB) (b) *Bus* (35.0240dB) (c) *BasketballDrill* (37.6760 dB) (d) *BQMall* (37.6272 dB)

(e) *Kimono* (39.9952dB) (f) *Tennis* (39.4329 dB) (g) *PeopleOnStreet* (37.6868dB) (h) *Traffic* (38.7295 dB)

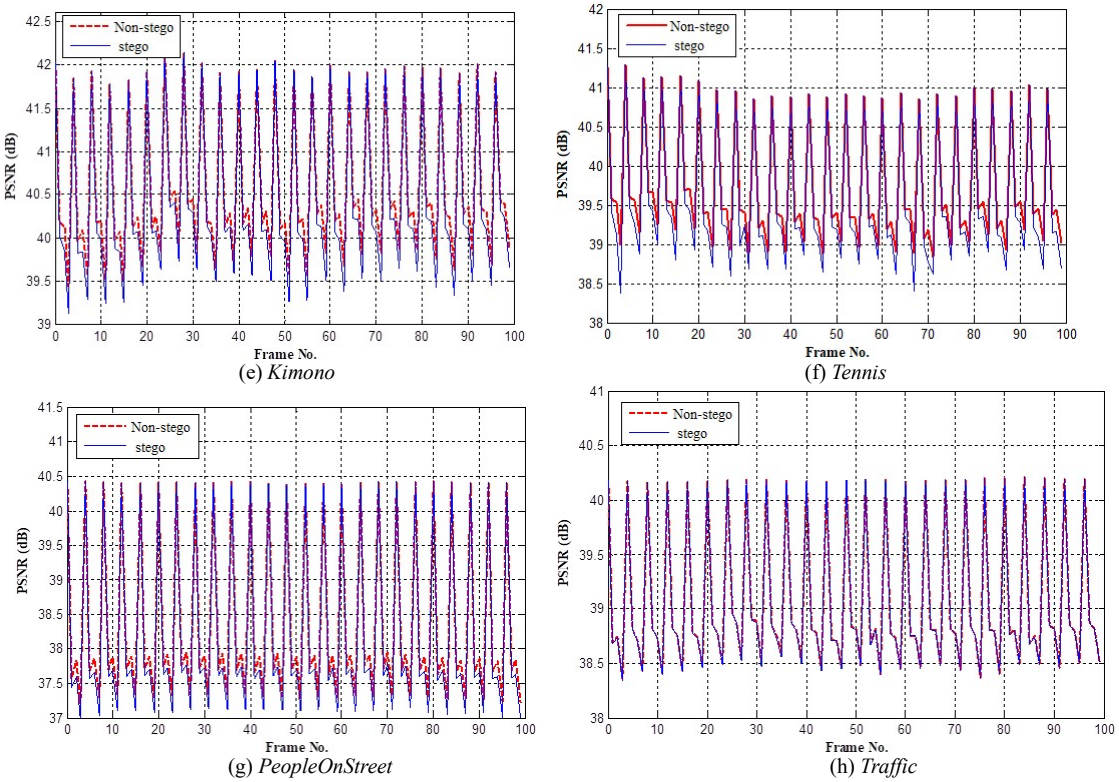
Fig.10 The corresponding decrypted video frames containing the hidden data



(a) *Football* (b) *Bus*

(c) *BasketballDrill* (d) *BQMall*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access



(e) *Kimono* (f) *Tennis*

(g) *PeopleOnStreet* (h) *Traffic*

Fig.11 *PSNR* values of all marked frames

C. Embedding Capacity

In TABLE.9, the maximum embedding capacity of each video is listed when *QP* is 28 and 32. It can be seen that the maximum embedding capacity is largely dependent on the video content. As shown in TABLE.9, the *Football*, *Kimono, and PeopleOnStreet* sequences provide significantly larger embedding capacity than other sequences. The reason is that the *Football* sequence has considerable motion, while *Kimono* and *PeopleOnStreet* sequences have high textures. Therefore, the number of *QTCs* satisfying the embedding condition in P-frames of these video sequences is relatively large. In contrast, the sequences *BasketballDrill*, *BQMall*, and *Traffic* are mostly static with limited motion in some frames and coded by *skip* blocks. In addition, the embedding capacity decreases with the increase of *QP* value. As the *QP* value increases, the number of residual coefficients decreases, and thus the qualified *bin-string* also decreases.

TABLE.9 TEST RESULTS OF THE PROPOSED SCHEME

Sequence	*QP*	*Maximum capacity* (*bits*)	*PSNR (dB)*		*SSIM*		*VQM*	
			Non-stego	*Stego*	*Non-stego*	*Stego*	*Non-stego*	*Stego*
Football	32	6177	34.2510	34.1064	0.9004	0.8998	1.1290	1.1898
	28	20379	37.1813	36.7557	0.9434	0.9421	0.8573	0.9682
Bus	32	1429	32.8120	32.7909	0.9328	0.9327	1.2732	1.2900
	28	8414	35.6082	35.5091	0.9606	0.9604	0.9743	1.0389
BasketballDrill	32	1832	35.6245	35.6135	0.9013	0.9013	1.0704	1.0803
	28	6040	37.8842	37.8499	0.9342	0.9341	0.8449	0.8795
BQMall	32	1913	35.2199	35.2087	0.9272	0.9271	1.0752	1.0831
	28	6766	37.5068	37.4703	0.9498	0.9497	0.8566	0.8982
Kimono	32	28554	38.6690	38.6086	0.9337	0.9334	0.9270	0.9570
	28	103359	40.5499	40.4079	0.9527	0.9522	0.7684	0.8186
Tennis	32	34480	37.9429	37.8276	0.9189	0.9184	0.9440	1.0038
	28	78172	39.7046	39.5438	0.9436	0.9431	0.7844	0.8697
PeopleOnStreet	32	69342	35.9914	35.9444	0.9261	0.9260	0.9739	1.0795
	28	229316	38.3239	38.1974	0.9498	0.9495	0.7723	0.8983
Traffic	32	2409	37.1660	37.1638	0.9399	0.9399	1.0257	1.0353
	28	8640	39.0604	39.0554	0.9581	0.9581	0.8420	0.8661

D. Bit Rate Overhead

In the proposed scheme, the encryption of *QTC*s sign bits, *MVD*s sign bits, and *MVD* absolute value suffixes does not affect the bit rate, because they are coded with the bypass coding mode wherein a fixed context is used. Moreover, data hiding is carried out by modifying a suitable *absCoeffLevel* to another *absCoeffLevel* which has the same length of *bin-string*. This has also been verified in our experiments.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2983330, IEEE Access

E. Discussion and Comparative Analysis

In previous works [18, 20-27], some efficient algorithms for embedding data into encrypted video are proposed. In H.264/AVC, when the absolute value of non-zero *QTCs* is not greater than 15, there are no suitable *bin-strings* available for information embedding [22, 23]. In HEVC, only the first two bins of the coefficient level are coded with context update. The remaining bins (i.e., *coeff_abs_level_remaining*) are coded in bypass mode [31]. Consequently, there are more qualified *bin-strings* in HEVC, which means a larger embedding capacity. The data hiding methods proposed in [24-26] are reversible, so it is not compared here. Compared with the method in [27], the proposed method has two advantages. First, encryption in [27] leads to an increase in video bit rate, while the proposed encryption method can keep the bit rate unchanged. Second, the embedding capacity is increased in most cases, as shown in TABLE.10. However, as the embedding capacity increases, the *PSNR* value will decrease correspondingly. Here, *ΔPSNR* represents the difference between the *PSNR* values before and after data embedding, and *BR_var* indicates the rate of video bit-rate increase caused by data embedding. In addition, the qualitative comparative analysis are also shown in TABLE.11. It can be seen that the proposed scheme is suitable for the latest video coding standard HEVC and keeps the bit rate unchanged.

TABLE.10 PERFORMANCE COMPARISON BETWEEN THE PROPOSED METHOD AND THE METHOD IN [27]

Sequence	*QP*	*Maximum capacity (bits)*		*ΔPSNR (dB)*		*BR_var*	
		Method in [27]	*Proposed method*	*Method in [27]*	*Proposed method*	*Method in [27]*	*Proposed method*
Football	32	2199	6177	0.0677	0.1446	3.88	0
	28	11138	20379	0.3447	0.4256	3.12	0
Bus	32	2834	1429	0.0582	0.0211	2.82	0
	28	26210	8414	0.3748	0.0991	1.71	0
BasketballDrill	32	890	1832	0.0074	0.011	18.02	0
	28	4358	6040	0.0429	0.0343	12.94	0
BQMall	32	1033	1913	0.0059	0.0112	9.11	0
	28	9338	6766	0.0533	0.0365	6.26	0
Kimono	32	15692	28554	0.0213	0.0604	5.19	0
	28	35412	103359	0.0335	0.142	4.23	0
Tennis	32	9594	34480	0.0186	0.1153	14.94	0
	28	14398	78172	0.03	0.1608	12.13	0

TABLE.11 COMPARATIVE ANALYSIS OF THE EXISTED SCHEMES

Methods	Elements for encryption	Elements for data embedding	Bit-rate increase	Separability	Reversibility	Coding standard
[18]	IPM, sign of MVD, sign of QTC	Amplitude of QTC	Yes	Yes	No	H.264/AVC
[20]	IPM, sign of MVD, sign of QTC	CAVLC codewords	No	Yes	No	H.264/AVC
[21]	IPM, sign of MVD, sign of QTC	CAVLC codewords	No	Yes	No	H.264/AVC
[22]	sign of MVD, sign of QTC	CABAC bin-strings	No	Yes	No	H.264/AVC
[23]	IPM, sign of MVD, sign of QTC	CABAC bin-strings	Yes	Yes	No	H.264/AVC
[24]	IPM, sign of MVD, sign of QTC	Amplitude of QTC	Yes	Yes	Yes	H.264/AVC
[25]	IPM, sign of MVD, sign of QTC	Amplitude of QTC	Yes	Yes	Yes	H.264/AVC
[26]	sign of MVD, amplitude of MVD, sign of QTC	Amplitude of QTC	Yes	Yes	Yes	HEVC
[27]	sign of MVD, sign of QTC, IPM	Amplitude of QTC	Yes	Yes	No	HEVC
Proposed method	**sign of MVD, amplitude of MVD, sign of QTC**	**Amplitude of QTC**	**No**	**Yes**	**No**	**HEVC**

IV. CONCLUSION AND FUTURE WORK

In this paper, an improved commutative encryption and data hiding scheme is presented, which can completely maintain the bit rate of HEVC video. In order to realize commutative property, one set of syntax elements is used for encryption, while another set of syntax elements is utilized for data hiding. The selective encryption is designed to encrypt *QTC*s sign bits, *MVD*s sign bits, and *MVD* absolute value suffixes, which has no effect on the HEVC video format compliance and bit rate. An improved coefficient modification technique is designed for data embedding, which results in a certain increase in embedding capacity. In addition, the data extraction process can be performed regardless of whether the video is in the encrypted or decrypted domain. The security analysis results also demonstrate that the scheme can achieve perceptual security and cryptographic security. Future work will focus on further improving the embedding rate and distortion performance of the algorithm.

Acknowledgements This work is supported by the National Natural Science Foundation of China (61771270), Zhejiang Provincial Natural Science Foundation of China (LR20F020001, LQ20F020025), and Ningbo Natural Science Foundation (2018A610054).

REFERENCES

[1] T. Stutz, A. Uhl. A, "A survey of H.264 AVC/SVC encryption," *IEEE Transactions on Circuits and Systems for Video Technology*, 2012, vol.22, no.3, pp. 325- 339.

[2] Y.Q. Tew, K. S. Wong, "An overview of information hiding in H.264/AVC compressed video," *IEEE Transactons on Circuits and Systems for Video Technology*, vol.24, no.2, pp. 305- 319, Feb. 2014.

[3] M Asikuzzaman, M R. Pickering. "An overview of digital video watermarking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.28, no.9, pp. 2131- 2153, Sept. 2018.

[4] Z. Shahid, M. Chaumont, W. Puech, "Fast protection of H.264/AVC by selective encryption of CAVLC and CABAC for I and P frames," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.21, no.5, pp. 565- 576, 2011.

[5] Y. S. Wang, M. O. Neill, F. Kurugollu, "A tunable encryption scheme and analysis of fast selective encryption for CAVLC and CABAC in H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 9, pp. 1476- 1490, 2013.

[6] B. Boyadjis, C. Bergeron, B. Pesquet-Popescu, F. Dufaux, "Extended Selective encryption of H.264/AVC (CABAC) and HEVC encoded video streams," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no.4, pp. 892-906, 2017.

[7] A. Sallam, O. Faragallah, E. EL-Rabaie, "HEVC selective encryption using RC6 block cipher technique," *IEEE Transactions on Multimedia*, vol.20, no.7, pp.1636-1644, Jul. 2018.

[8] R. J. Mstafa, K.M. Elleithy, E. Abdelfattah, "A robust and secure video steganography method in DWT-DCT domains based on multiple object tracking and ECC," *IEEE Access*, vol. 5, pp. 5354-5365, Apr. 2017.

[9] H. Mareen, J. Praeter, G. V. Wallendael, P. Lambert, "A scalable architecture for uncompressed-domain watermarked videos," *IEEE Transactions on Information Forensics and Security*, vol.14, no.6, pp. 1432-1444, Jun.2019.

[10] D.W. Xu, R.D. Wang, J.C. Wang, "Prediction mode modulated data-hiding algorithm for H.264/AVC," *Journal of Real-Time Image Processing*, vol.7, no.4, pp.205-214, Dec. 2012.

[11] H. A.Aly, "Data hiding in motion vectors of compressed video based on their associated prediction error," *IEEE Transactions on Information Forensics and Security*, vol.6, no.1, pp. 14-18, Mar. 2011.

[12] Y.X. Liu, M.S. Hu, X.J. Ma, H.G. Zhao, "A new robust data hiding method for H.264 AVC without intra frame distortion drift," *Neurocomputing*, vol.151, pp. 1076-1085, Mar. 2015.

[13] T.Stütz, F.Autrusseau, A. Uhl, "Non-blind structure-preserving substitution watermarking of H.264/CAVLC inter-frames," *IEEE Transactions on Multimedia*, vol.16, no.5, pp. 1337 – 1349, Aug. 2014.

[14] P. C. Chang, K. L. Chung, J. J. Chen, et al, "A DCT/DST-based error propagation-free data hiding algorithm for HEVC intra-coded frames," *Journal of Visual Communication and Image Representation*, vol.25, no.2, pp.239–253, Feb. 2014.

[15] T. Dutta, H. P. Gupta, "A robust watermarking framework for High Efficiency Video Coding (HEVC) – Encoded video with blind extraction process," *Journal of Visual Communication and Image Representation*, vol.38, pp. 29-44, Jul.2016.

[16] T. Dutta, H. P. Gupta, "An efficient framework for compressed domain watermarking in P frames of high-efficiency video coding (HEVC)--encoded video," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol.13, no.1, pp.2-24, Jan. 2017.

[17] J. Yang, S. B. Li, "An efficient information hiding method based on motion vector space encoding for HEVC," *Multimedia Tools and Applications*, vol.77. no.10, pp. 11979–12001, May.2018.

[18] S. G. Lian, Z. X. Liu, Z. Ren, "Commutative encryption and watermarking in video compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.17, no.6, pp.774 – 778, Jun. 2007.

[19] S.G. Lian, "Quasi-commutative watermarking and encryption for secure media content distribution," *Multimedia Tools and Applications*, vol.43, no.1, pp.91-107, May.2009.

[20] D.W. Xu, R.D. Wang, Y. Q.Shi, "Data hiding in encrypted H.264/AVC video streams by codeword substitution," *IEEE Transactions on Information Forensics and Security*, vol.9, no.4, pp.596-606, Apr.2014.

[21] D.W. Xu, R.D. Wang, Y. Q.Shi, "An Improved Scheme for Data Hiding in Encrypted H.264/AVC Videos," *Journal of Visual Communication and Image Representation*, vol.36, pp. 229-242, Apr.2016.

[22] D.W. Xu, R.D. Wang, "Context adaptive binary arithmetic coding-based data hiding in partially encrypted H.264/AVC videos, " *Journal of Electronic Imaging*, vol.24, no.3, 033028:1-13, May. 2015.

[23] D.W. Xu, R.D. Wang, Y. N. Zhu, "Tunable data hiding in partially encrypted H.264/AVC videos, " *Journal of Visual Communication and Image Representation*, vol.45, pp. 34-45, 2017.

[24] D.W. Xu, R.D. Wang, "Efficient reversible data hiding in encrypted H.264/AVC Videos," *Journal of Electronic Imaging*, vol.23, no.5, 053022:1-14, Sep. 2014.

[25] Y. Z. Yao, W. M. Zhang, N. H. Yu, "Inter-frame distortion drift analysis for reversible data hiding in encrypted H.264AVC video bitstreams," *Signal Processing*, vol.128, pp. 531-545, Nov. 2016.

[26] M. Long, F. Peng, H.Y. Li, "Separable reversible data hiding and encryption for HEVC video," *Journal of Real-time Image Processing*, vol.14, no.1, pp.171-182, Jan.2018.

[27] Dawen Xu, "Commutative encryption and data hiding in HEVC video compression," *IEEE Access*, vol.7, pp. 66028 – 66041, May 2019.

[28] X. Zhang, "Commutative Reversible Data Hiding and Encryption," *Security and Communication Networks*, vol.6, pp. 1396-1403, Mar. 2013.

[29] D. Marpe, H. Schwarz, T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.13, no.7, pp. 620 – 636, Jul. 2003.

[30] G.J. Sullivan, J.R. Ohm, W.J. Han, T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and System for Video Technology*, vol.22, no.12, pp. 1649-1668, Dec.2012.

[31] V. Sze, M. Budagavi, "High throughput CABAC entropy coding in HEVC," *IEEE Transactions on Circuits and System for Video Technology*, vol.22, no.12, pp. 1778-1791, Dec. 2012.

[32] J. Sole, R. Joshi, N. Nguyen, etc., "Transform coefficient coding in HEVC," *TEEE Transactions on Circuits and Systems for Video Technology*, vol.22, no.12, pp. 1765-1777, Dec 2012.

[33] HEVC Reference Software HM 12.0 [Online]. Available: https://hevc.hhi.fraunhofer.de/trac/hevc/browser/tags